Q1. ***Using the browser console***
1. Open the browser console, type 4 + 10, and press Enter. What do you see as the response?
2. Use the console.log() syntax, placing a value within the rounded brackets.
Try entering your name with quotes around it (this is to indicate the fact that it's a text string).

Q2. **Creating an HTML file and a linked JavaScript file**
Create an HTML file and create a separate JavaScript file. Then, connect to the
JavaScript file from the HTML file.
1. In the JavaScript file, output your name into the console and add a multiple line
comment to your code.
2. Try commenting out the console message in your JavaScript file so that nothing shows in the console.

Q3. **Write some code to calculate the hypotenuse of a triangle using the Pythagorean**
theorem when given the values of the other two sides. The theorem specifies that the
relation between the sides of a right-angled triangle is a2 + b2 = c2.
You can use prompt() to get the value for a and b.

Q4 Create variables for three numbers: a, b, and c. Update these variables with the
following actions using the assignment operators:
• Add b to a
• Divide a by c
• Replace the value of c with the modulus of c and b
• Print all three numbers to the console

Q5.Miles-to-kilometers converter
Create a variable that contains a value in miles, convert it to kilometers, and log the
value in kilometers in the following format:

The distance of 130 kms is equal to 209.2142 miles
For reference, 1 mile equals 1.60934 kilometers.

Q6. BMI calculator
Set values for height in inches and weight in pounds, then convert the values to
centimeters and kilos, outputting the results to the console:
• 1 inch is equal to 2.54 cm
• 2.2046 pounds is equal to 1 kilo
Output the results. Then, calculate and log the BMI: this is equal to weight (in kilos)
divided by squared height (in meters). Output the results to the console.

Q7.Create an array to use as your shopping list with 3 items: "Milk," "Bread," and
"Apples."
2. Check your list length in the console.
3. Update "Bread" to "Bananas."
4. Output your entire list to the console.

Q8
1. Create an empty array to use as a shopping list.
2. Add Milk, Bread, and Apples to your list.
3. Update "Bread" with Bananas and Eggs.
4. Remove the last item from the array and output it into the console.
5. Sort the list alphabetically.
6. Find and output the index value of Milk.
7. After Bananas, add Carrots and Lettuce.
8. Create a new list containing Juice and Pop.
9. Combine both lists, adding the new list twice to the end of the first list.
10. Get the last index value of Pop and output it to the console.
11. Your final list should look like this:
["Bananas", "Carrots", "Lettuce", "Eggs", "Milk", "Juice", "Pop", "Juice", "Pop"]

Q9

1. Create an array containing three values: 1, 2, and 3.
2. Nest the original array into a new array three times.
3. Output the value 2 from one of the arrays into the console.

Q10

1. Create a new myCar object for a car. Add some properties, including, but not
limited to, make and model, and values for a typical car or your car. Feel free
to use booleans, strings, or numbers.
2. Create a variable that can hold the string value color. This variable containing a string value color can now be used to reference the property name within myCar. Then, use the variable within the square bracket notation
to assign a new value to the color property in myCar.
3. Use that same variable and assign a new property string value to it, such as
forSale. Use the bracket notation once again to assign a new value to the forSale property to indicate whether the car is available for purchase.
4. Output make and model into the console.
5. Output the value of forSale into the console.

Q11

1. Create an object named people that contains an empty array that is called
friends.
2. Create three variables, each containing an object, that contain one of your
friend's first names, last names, and an ID value.
3. Add the three friends to the friend array.
4. Output it to the console.

Q12

Manipulating an array

Take the following array:
const theList = ['Laurence', 'Svekis', true, 35, null, undefined,
{test: 'one', score: 55}, ['one', 'two']];
Manipulate your array using various methods, such as pop(), push(), shift(), and
unshift(), and transform it into the following:
["FIRST", "Svekis", "MIDDLE", "hello World", "LAST"]
You can take the following steps, or adopt your own approach:
• Remove the first item and the last item.
• Add FIRST to the start of the array.
• Assign hello World to the fourth item value.
• Assign MIDDLE to the third index value.
• Add LAST to the last position in the array.
• Output it to the console.

Q13
Company product catalog
In this project, you will implement a data structure for a product catalog and create
queries to retrieve data.
1. Create an array to hold an inventory of store items.
2. Create three items, each having the properties of name, model, cost, and quantity.
3. Add all three objects to the main array using an array method, and then log
the inventory array to the console.
4. Access the quantity element of your third item, and log it to the console.
Experiment by adding and accessing more elements within your data structure.

Q14.
1. Create a variable with a Boolean value.
2. Output the value of the variable to the console.
3. Check whether the variable is true and if so, output a message to the console,
using the following syntax:
if(myVariable){

//action
}
4. Add another if statement with an ! in front of the variable to check whether
the condition is not true, and create a message that will be printed to the console in that instance. You should have two if statements, one with an ! and the other without. You could also use an if and an else statement instead—experiment!
5. Change the variable to the opposite to see how the result changes.

Q15
1. Create a prompt to ask the user's age
2. Convert the response from the prompt to a number
3. Declare a message variable that you will use to hold the console message for
the user
4. If the input age is equal to or greater than 21, set the message variable to confirm entry to a venue and the ability to purchase alcohol
5. If the input age is equal to or greater than 19, set the message variable to confirm entry to the venue but deny the purchase of alcohol
6. Provide a default else statement to set the message variable to deny entry if
none are true
7. Output the response message variable to the console

Q16
1. Create a Boolean value for an ID variable
2. Using a ternary operator, create a message variable that will check whether
their ID is valid and either allow a person into a venue or not
3. Output the response to the console

Q17 create a Magic 8-Ball random answer generator:
1. Start by setting a variable that gets a random value assigned to it. The value is assigned by generating a random number 0-5, for 6 possible results. You can increase this number as you add more results.
2. Create a prompt that can get a string value input from a user that you can repeat back in the final output.

3. Create 6 responses using the switch statement, each assigned to a different value from the random number generator.
4. Create a variable to hold the end response, which should be a sentence printed for the user. You can assign different string values for each case, assigning new values depending on the results from the random value.
5. Output the user's original question, plus the randomly selected case response, to the console after the user enters their question.

Q18

1. Create a variable called prize and use a prompt to ask the user to set the value by selecting a number between 0 and 10
2. Convert the prompt response to a number data type
3. Create a variable to use for the output message containing the value "My Selection: "
4. Using the switch statement (and creativity), provide a response back regarding a prize that is awarded depending on what number is selected
5. Use the switch break to add combined results for prizes
6. Output the message back to the user by concatenating your prize variable
strings and the output message string

Q19

Friend checker game
Ask the user to enter a name, using the switch statement to return a confirmation
that the user is a friend if the name selected is known in the case statements. You
can add a default response that you don't know the person if it's an unknown name.
Output the result into the console.

Q20

Rock Paper Scissors game

This is a game between a player and the computer, where both will make a random selection of either Rock, Paper, or Scissors (alternatively, you could create a version using real player input!). Rock will beat out Scissors, Paper will beat out Rock, and Scissors will beat out Paper. You can use JavaScript to create your own version of this game, applying the logic with an if statement. Since this project is a little more difficult, here are some suggested steps:

1. Create an array that contains the variables Rock, Paper, and Scissors.

2. Set up a variable that generates a random number 0-2 for the player and then do the same for the computer's selection. The number represents the index values in the array of the 3 items.

3. Create a variable to hold a response message to the user. This can show the random results for the player and then also the result for the computer of the matching item from the array.

4. Create a condition to handle the player and computer selections. If both are the same, this results in a tie.

5. Use conditions to apply the game logic and return the correct results. There are several ways to do this with the condition statements, but you could check which player's index value is bigger and assign the victory accordingly, with the exception of Rock beating Scissors.

6. Add a new output message that shows the player selection versus the computer selection and the result of the game.

Q21

1. Create a variable to be used as the max value for the number guessing game.

2. Generate a random number for the solution using Math.random() and Math.floor(). You will also need to add 1 so that the value is returned as 1-[whatever the set max value is]. You can log this value to the console for

development to see the value as you create the game, then when the game is complete you can comment out this console output.
3. Create a variable that will be used for tracking whether the answer is correct or not and set it to a default Boolean value of false. We can update it to be true if the user guess is a match.
4. Use a while loop to iterate a prompt that asks the user to enter a number between 1 and 5, and convert the response into a number in order to match the data type of the random number.
5. Inside the while loop, check using a condition to see if the prompt value is equal to the solution number. Apply logic such that if the number is correct, you set the status to true and break out of the loop. Provide the player with some feedback as to whether the guess was high or low, and initiate another prompt until the user guesses correctly. In this way we use the loop to keep asking until the solution is correct, and at that point we can stop the iteration of the block of code.

Q22
create a basic counter that will increase a dynamic variable
by a consistent step value, up to an upper limit.
1. Set the starting counter to 0
2. Create a variable, step, to increase your counter by
3. Add a do while loop, printing the counter to the console and incrementing it by the step amount each loop
4. Continue to loop until the counter is equal to 100 or more than 100

Q23
1. Setup a blank array, myWork.
2. Using a for loop, create a list of 10 objects, each of which is a numbered lesson (e.g. Lesson 1, Lesson 2, Lesson 3….) with an alternating true/false status for every other item to indicate whether the class will be running this year. For example:
name: 'Lesson 1', status: true
3. You can specify the status by using a ternary operator that checks whether the modulo of the given lesson value is equal to zero and by setting up a Boolean value to alternate the values each iteration.
4. Create a lesson using a temporary object variable, containing the name (lesson with the numeric value) and predefined status (which we set up in the previous step).
5. Push the objects to the myWork array.
6. Output the array to the console.

Q24

Generating a table of values

1. To create a table generator, first create an empty array, myTable, to hold your table data.

2. Set variable values for the number of rows and columns. This will allow us to dynamically control how many rows and columns we want within the table. Separating the values from the main code helps make updates to the dimensions easier.

3. Set up a counter variable with an initial value of 0. The counter will be used to set the content and count the values of the cells within the table.

4. Create a for loop with conditions to set the number of iterations, and to construct each row of the table. Within it, set up a new temporary array (tempTable) to hold each row of data. The columns will be nested within the rows, generating each cell needed for the column.

5. Nest a second loop within the first to count the columns. Columns are run within the row loop so that we have a uniform number of columns within the table.

6. Increment the main counter each iteration of the inner loop, so that we track a master count of each one of the cells and how many cells are created.

7. Push the counter values to the temporary array, tempTable. Since the array is a nested array representing a table, the values of the counter can also be used to illustrate the cell values next to each other in the table. Although these are separate arrays representing new rows, the value of the counter will help illustrate the overall sequence of cells in the final table.

8. Push the temporary array to the main table. As each iteration builds a new row of array items, this will continue to build the main table in the array.

9. Output into the console with console.table(myTable). This will show you a visual representation of the table structure.

Q25.

Create a table grid that contains nested arrays as rows within a table.

The rows will each contain the number of cells needed for the number of columns set in the variables. This grid table will dynamically adjust depending on the values for the variables.

1. Create a grid array variable.

2. Set a value of 64 for the number of cells.

3. Set a counter to 0.

Don't worry, we will cover this function and many more in detail in Chapter 8, Built-in JavaScript Methods.

4. Create a global variable to be used for the row array.

5. Create a loop that will iterate up to the number of cells you want in the array, plus one to include the zero value. In our example, we would use 64+1.

6. Add an outer if statement, which uses modulo to check if the main counter is divisible by 8 or whatever number of columns you want.

7. Inside the preceding if statement, add another if statement to check if the

row is undefined, indicating whether it is the first run or whether the row is complete. If the row has been defined, then add the row to the main grid array.
8. To finish off the outer if statement, if the counter is divisible by 8, clear the row array—it has already been added to the grid by the inner if statement.
9. At the end of the for loop, increment of the main counter by 1.
10. Set up a temporary variable to hold the value of the counter and push it to the row array.
11. Within the loop iteration, check if the value of the counter is equal to the total number of columns you want; if it is, then add the current row to the grid.
12. Please note that the extra cell will not be added to the grid since there aren't enough cells to make a new row within the condition that adds the rows to the grid. An alternative solution would be to remove the +1 from the loop condition and add grid.push(row) after the loop is completed, both of which will provide the same solution output.
13. Output the grid into the console.

Q26
1. Create a simple object with three items in it.
2. Using the for in loop, get the properties' names and values from the object and output them into the console.
3. Create an array containing the same three items. Using either the for loop or the for in loop, output the values from the array into the console.

Q27
Math multiplication table
In this project, you will create a math multiplication table using loops. You can do this using your own creativity or by following some of the following suggested steps:
1. Set up a blank array to contain the final multiplication table.
2. Set a value variable to specify how many values you want to multiply with one another and show the results for.
3. Create an outer for loop to iterate through each row and a temp array to store the row values. Each row will be an array of cells that will be nested into the final table.
4. Add an inner for loop for the column values, which will push the multiplied row and column values to the temp array.
5. Add the temporary row data that contains the calculated solutions to the main array of the final table. The final result will add a row of values for the calculations.